2006

Streaming in MANET: Proactive Link Protection and Receiver-Oriented Adaptation

Toby Xu
Iowa State University

Ying Cai
Iowa State University, yingcai@iastate.edu

Streaming in MANET: Proactive Link Protection and Receiver-Oriented Adaptation

Abstract
Multimedia streaming applications can signifincantly boost the value of mobile ad hoc networks (MANET). Live streaming, however, means continuous data delivery, which is a major challenge in MANET. Because of host mobility, a streaming path may be broken easily, causing streaming interruption. In this paper, we address this problem with a light-weighted yet robust streaming protocol. Our technique distinguishes itself from existing ones with two innovative features: proactive link protection (PLP) and receiveroriented adaptation (ROA). PLP allows a mobile host in a streaming path to find an alternative link before its current one becomes broken. This feature minimizes the chance of having to discover a new path in urgent to replace a broken link. While PLP protects the streaming continuity, ROA ensures the streaming efficiency by minimizing the hop number of a streaming path. Specifically, ROA dynamically adjusts the path of a live stream to make it as straight as possible along the direction from the source to the receiver. We evaluate the proposed technique through simulation, and our extensive performance study indicates that the new technique can support robust streaming applications with a minimal control overhead.

Keywords
MANET, live streaming, route maintenance

Disciplines
OS and Networks

Streaming in MANET: Proactive Link Protection and Receiver-Oriented Adaptation

Toby Xu *Ying Cai*

Department of Computer Science
Iowa State University
Ames, Iowa 50011, U.S.A.
E-mail: {tobyxu, yingcai}@cs.iastate.edu

Abstract

Multimedia streaming applications can signifincantly boost the value of mobile ad hoc networks (MANET). Live streaming, however, means continuous data delivery, which is a major challenge in MANET. Because of host mobility, a streaming path may be broken easily, causing streaming interruption. In this paper, we address this problem with a light-weighted yet robust streaming protocol. Our technique distinguishes itself from existing ones with two innovative features: proactive link protection *(PLP) and* receiver-oriented adaptation *(ROA). PLP allows a mobile host in a streaming path to find an alternative link before its current one becomes broken. This feature minimizes the chance of having to discover a new path in urgent to replace a broken link. While PLP protects the streaming continuity, ROA ensures the streaming efficiency by minimizing the hop number of a streaming path. Specifically, ROA dynamically adjusts the path of a live stream to make it as straight as possible along the direction from the source to the receiver. We evaluate the proposed technique through simulation, and our extensive performance study indicates that the new technique can support robust streaming applications with a minimal control overhead.*

KEYWORDS: *MANET, live streaming, route maintenance.*

1 Introduction

A mobile ad hoc network (MANET) consists of a set of mobile devices that communicate with each other through wireless packet relaying. Since it does not rely on any fixed communication infrastructure, a MANET can be set up quickly at a minimal cost. This feature makes it highly attractive to applications such as military operations and disaster recovery, which demand a self-sustain and quickly deployable network. With continuous price drop and miniaturization of electronic components, many mobile devices are now multimedia-capable, making it possible for more informative communication in MANET. For instance, soldiers in battlefield may videotape their surrounding environment and stream the live data in real time to their commanders. Obviously, such live streaming applications make MANET much more valuable.

Live streaming means continuous data delivery. To guarantee a smooth playback, each data packet must be delivered with a strict deadline. This is a major challenge for routing protocols in MANET, where the network topology is ever changing while a data packet usually needs to go through many hops before reaching its recipient. Because of host mobility, a routing path used by the current data packet may become unavailable for the next one. Existing routing protocols, either proactive or reactive, are not unsuitable for live streaming applications. In protocols such as DSDV [1], each mobile host maintains a routing table and periodically broadcasts such information to other hosts in the network. Such proactive route discovery minimizes the time of setting up a streaming

path, but may generate a large amount of control overhead, which itself can constitute a significant portion of network traffic. Reactive routing protocols such as DSR [2] and AODV [3] avoid this problem by discovering a routing path at the time when a data packet needs to be delivered. For performance improvement, the discovered routes can be cached for future data delivery. However, when an existing link is found broken, a new route needs to be discovered on the fly, which may cause long delay and interruption of continuous data delivery. In case of live streaming, a streaming path may need to be re-constructed frequently due to host mobility, resulting in a poor playback quality at the receiver side.

To address the above problem, some multi-path routing protocols are proposed (e.g., [4], [5], [6], [7], etc.). For between each pair of source and destination, these schemes construct multiple disjoint paths and chooses one of them as a working path for data delivery. When the working path breaks, the data flow switches to another backup path. Unfortunately, this strategy can not eliminate the problem of path reconstruction either, because the backup paths may also become broken as the network topology changes. Finding multiple paths also incurs more overhead in path discovery and in particular, may be impossible when the network is not dense enough.

In this paper, we present a light-weighted yet highly robust streaming protocol for MANET. Unlike multi-path approaches, our protocol constructs only a single path. This path is then maintained using two innovative techniques, *proactive link protection* (PLP) and *receiver-oriented adaptation* (ROA). The idea of PLP is to allow each host in the streaming path to monitor the robustness of its connection, and proactively looks for a backup link when the current connection is about to break. Since the hosts can simultaneously look for backup links, this scheme allows a stream to continue to flow even in the presence of multiple link breakages. While PLP minimizes the chance of streaming interruption, the second technique, ROA, makes it possible to maintain a high-quality streaming path. A routing path may be of high quality (e.g., having a minimum number of hops) at the time when it is discovered; but as the time goes, the path usually deteriorates and will eventually need to be reconstructed. ROA addresses this problem by dynamically adjusting a streaming path to make it as straight as possible along the direction from the stream source to the receiver. We evaluate the proposed techniques through simulation, and our extensive performance study indicates that the new technique can be used for robust and efficient live streaming.

The remainder of this paper is organized as follows. In Section 2, we present our streaming protocol in detail. The preformance of the proposed technique is evaluated in Section 3. In Section 4, we review more related work and in Section 5, we conclude this paper.

2 Proposed Streaming Protocol

We assume all mobile hosts move in a planar area, and each of them has a unique ID. We also assume that all mobile hosts are position-aware (e.g., GPS-enabled), and they have the same transmission range. Unlike many other techniques, we do not require each host to track its neighbors through periodic heartbeat, which may incur significant control overhead. Thus, a host is invisible to its neighbors unless it is transmitting data packets. On the other hand, a host sending data is known to all of its neighbors, since each host can eavesdrop any on-going communication within its own wireless coverage.

To start a live stream, the source first discovers a routing path to its receiver. This can be done by using any existing route discovery algorithm such as AODV or DSR. The source then starts to send data using the discovered path. For each data packet, the following fields are included in its header:

- *source_pos*: the position of the source when it sends out this packet;
- *hop_cnt*: the number of hops that this packet has traversed through, it is initially set to be 0 and incremented each time the packet is rebroadcast;
- *up_id, up_pos*: the ID and the position of the intermediate host which forwards this packet;
- *dest_pos*: the position of the destination.

When a host forwards a data packet, it may need to adjust the values of some of these fields. For instance, the hop_cnt of a packet is initially set to be 0 and incremented each time when the packet is rebroadcast. In addition, the field of $dest\_pos$ is used for piggybacking the location of the destination host. Specifically, when the destination host receives a packet, it includes

its current position in its ACK packet. This position information is then piggybacked upward to the source during the data transmission.

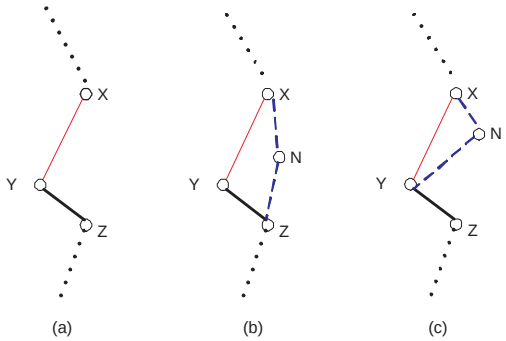
Because of host mobility, the initial streaming path can be broken, resulting in streaming interruption. In the next subsections, we address this problem with two new techniques.

This technique is aimed at replacing a link that is about to break by proactively looking for an alternative one. Figure 1(a) shows a link $\overline{XY}$ in a streaming path. When X sends a data packet to Y, Y will either rebroadcast the packet or respond with an ACK (in this case, Y is the destination host). By eavesdropping the data sent by Y, X can estimate the link's *rest life time*, denoted as $RLT_{\overline{XY}}$. For the purpose of such estimation, various techniques can be used. For instances, X can use Y's position information to calculate their relative movement, monitor the variation of Y's signal strength, or use other measurement techniques such as ETX [8]. We say a link is in critical if its rest life time is less than some threshold. In Figure 1, the thin solid line denotes the critical link $\overline{XY}$, and the thick solid line denotes the healthy link $\overline{YZ}$. When X determines that its connection to Y is in critical, X broadcasts a message $REPAIR(X, Y)$ locally (i.e., within one hop). When a host, say N, receives such a message, it checks if it can provide an alternative route to backup $\overline{XY}$. There are two kinds of backup routes N can find:

- Case 1: N is a 1-hop neighbor to Y's downstream host, say Z. In this case, N can be a *replacement host*, i.e., replacing Y to relay data from X to Z.
- Case 2: N is within 1-hop distance to Y. In this case, N can serve as a *brigding host* to forward data from X to Y.

The above two cases are illustrated in Figure 1(b) and (c), respectively. In the first case, N estimates $RLT_{\overline{NX}}$ and $RLT_{\overline{NZ}}$, and then responds with a message $REPLACE(N, t)$, where t is set to be $min(RLT_{\overline{NX}}, RLT_{\overline{NZ}})$ (i.e., the estimated RLT for path $X \rightarrow N \rightarrow Z$). In the second case, N responds with a message $BRIDGE(N, t)$, where t is equal to $min(RLT_{\overline{NX}}, RLT_{\overline{NY}})$.

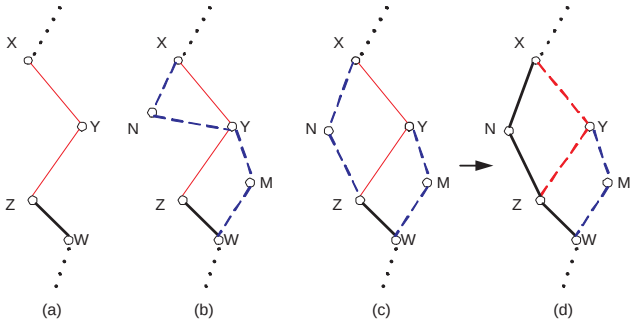
When X receives a message $REPLACE(N, t)$, it replaces Y with N and from then on, forwards all data to N. X may receive more then one REPLACE message. When this happens, X chooses the replacement host with the longest estimated RLT. Note that if a host can be found to replace Y, the hop number of a streaming path will not increase. If X does not receive any $REPLACE$ message, it checks all $BRIDGE$ messages it receives and chooses the bridging host with the longest estimated RLT. If X receives neither $REPLACE$ nor $BRIDGE$ messages, X sends a message $ERROR(X, Y)$ to its upstream host, which will try to find an alternative link like as X does. When the $ERROR$ message reaches the source, a route rediscovery is initiated.



In the above scheme, each host in a streaming path can perform its own link protection independently. This fully decentralized protection makes it possible for a stream to survive in the presence of multiple link breakages, even when these breakages are consecutive. Consider Figure 2(a), which shows a streaming path $X \rightarrow Y \rightarrow Z \rightarrow W$. Suppose links $\overline{XY}$ and $\overline{YZ}$ are critical. For their own link protections, both X and Y will be simulatenously looking for new hosts to construct alternative routes. Regardless of what new links constructed by X and Y, the stream can always continue to flow. Suppose N is the host selected by X to be added in the streaming path, we consider the following two scenarios:

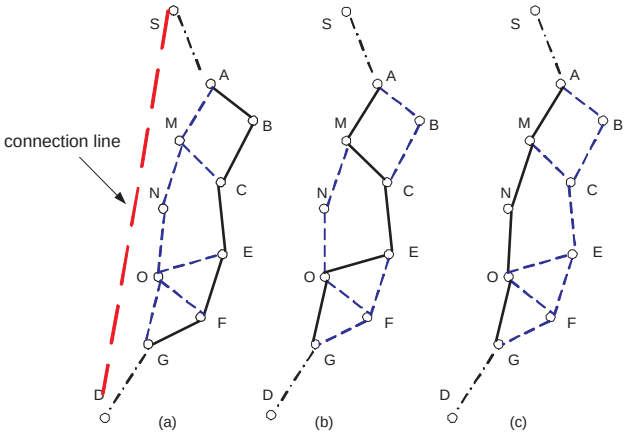
- Case 1: N is a bridging host. That is, $X \rightarrow N \rightarrow Y$, as shown in Figure 2(b). As N connects to Y and Y must be in the streaming path regardless of what new links found by Y, the path continuity is guaranteed.
- Case 2: N is a replacement host. That is, Y is replaced by N and $X \rightarrow N \rightarrow Z$, where Z is Y's current downstream node. Since the stream flows

through $X \rightarrow N \rightarrow Z$ and Z connects to the remaining path that leads to the stream's destination, the path continuity is also gunranteed. Note that Y may choose another host, say M, to protect its own critical link $\overline{YZ}$. However, since Y is replaced by N, neither Y nor M will receive data packets. As a result, they can automatically remove themselves from the streaming path. This scenario is illustrated in Figure 2 (c) and (d). .



The initial route discovered by techniques such as DSR or AODV is likely to be the shortest. However, this route will become longer and longer as the hosts in the path try to maintain the connection from the source to the destination. In PLP, for instance, whenever a host constructs an alternative link using a bridging host, it adds one more hop to the routing path. A longer path leads to a higher end-to-end delay and also consume more networking overheads. In this section, we address this problem.

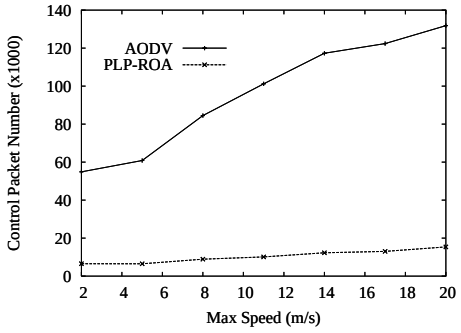
Recall that PLP does not require each host to periodically broadcast its heartbeat. Instead, each host simply needs to eavesdrop the on-going communication among its neighbors. Suppose a host, say N, is a neighbor of two hosts, say X and Y, which are transmitting data packets for a live stream. N can compare the field values of *hop_cnt* in the data packets from transmitted by X and Y. If the difference of these two values is larger than 2, then N can shorten the streaming path by bridging the communication between X and Y, i.e., $X \rightarrow N \rightarrow Y$. Although this adaptation is simple to implement, a host cannot bridge an existing path unless it can detect by itself that it can reduce the hop number by at least one. Unfortunately, many times a shorter path does exist but can not be found locally (i.e., within 1-hop). As an example, consider Figure 3(a). Each solid line means an active link and they together form the current streaming path from source S to destination D. On the other hand, each dash line connecting two hosts means they are 1-hop neighbors to each other. It is clear that no host can detect by itself that it can shorten this streaming path. However, there does exist a shorter path from A to G, i.e., $A \rightarrow M \rightarrow N \rightarrow O \rightarrow G$.



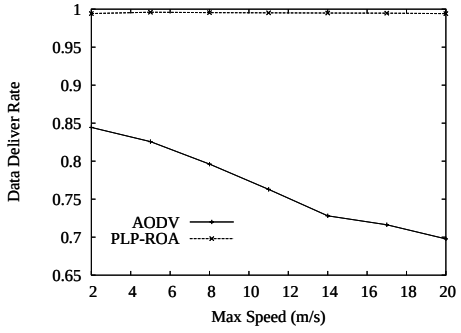
To shorten a streaming path more effectively, we propose the following technique, which we refer to as receiver-oriented adaptation (ROA). Our goal is to make a streaming path as straight as possible along the connection line from the source to its destination. Since each data packet contains the *source_pos* and the *dest_pos*, this connection line can be determined by the hosts that are within 1-hop distance to any host in the streaming path. When a host can provide an alternative link for the streaming path, it first checks if the alternative link is closer to the connection line than the existing link. If this is true, the host joins the streaming path by replacing the existing link. This approach allows a streaming path to be gradually adjusted to the shortest path, although not every adjustment can shorten the streaming path. As an example, consider Figure 3(a) again. The alternative path $A \rightarrow M \rightarrow C$ is closer to $\overline{SD}$ but not shorter than the existing path $A \rightarrow B \rightarrow C$. Similarly, $E \rightarrow O \rightarrow G$ is closer to $\overline{SD}$ but not shorter than $E \rightarrow F \rightarrow G$. However, after links $A \rightarrow M \rightarrow C$ and $E \rightarrow O \rightarrow G$ replace $A \rightarrow B \rightarrow C$ and $E \rightarrow F \rightarrow G$, respectively, host

N, which is not currently in the streaming path, can then detect a shorter path between M and O. Thus, the streaming path $A \rightarrow B \rightarrow C \rightarrow E \rightarrow F \rightarrow G$ is eventually replaced by $A \rightarrow M \rightarrow N \rightarrow O \rightarrow G$.

The above discussion implicitly assumes that the host that can provide an alternative link knows the status of the existing link. In reality, this may always be true. For example, in Figure 3, host M is two-hop away from B and therefore, has no idea about B's position. Thus, M can not compare the alternate link with the existing link between B and A. Fortunately, this problem is addressable. Assuming that all mobile hosts have the same transmission range, we can prove that if both B and M are 1-hop neighbors to A and C, and located on the same side of the line $\overline{AC}$, then B and M must be 1-hop neighbor to each other. Thus, if M does not know B's position, these two hosts must be on the different sides of line $\overline{AC}$. In this case, M can first compute the distances from its own, A and C to the connection line, respectively, and then compare these three distances. If M is closer to the connection line than A and C, M must also be closer to the connection line than B. Thus, M can determine if it needs to join the streaming path.



(a) control overhead



(b) data delivery ratio

3. Performance Study

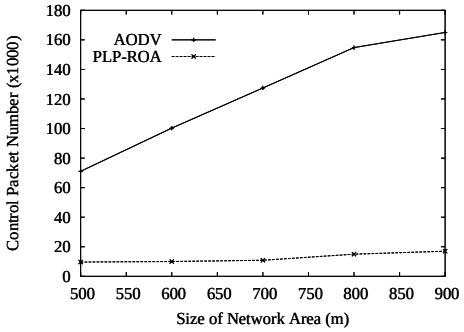
To evaluate the efficiency and effectiveness of the proposed schemes, we have implemented a detailed simulator. We compare the proposed techniques against AODV. Since AODV has been used as the baseline in many performance studies, comparing with AODV makes it possible to compare our techniques indirectly with other existing techniques.

In our simulation, the transmission range of every mobile host is set to be 100 meters. For each simulation, we generated a number of mobile hosts and randomly deployed them in the network domain. The speed of each mobile host is randomly chosen from 1 to MAX_SPEED meters per second, and remains constant during simulation. Their initial moving directions are set randomly. Each host moves linearly until it reaches the boundary of the network domain and when this happens, it reflects at the boundary and moves with the same speed. In each simulation, we start 10 live streams, each having a pair of randomly chosen source and destination. The start time and duration of each streaming task are also randomly selected from the simulation period. Finally, with the proposed techniques, a host in a streaming path will search for a protection path whenever it determines that its current link's RFT is less than 2 second.

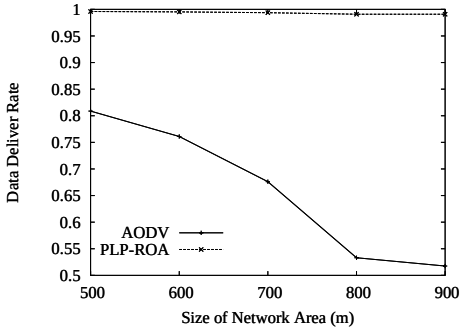
In this study, we investigated the effect of host mobility on the performance of AODV and PLP-ROA in terms of control overhead and data delivery rate. We generated 400 mobile hosts and randomly deploy them on an area of $600 \times 600\ meter^2$. The MAX_SPEED of mobile hosts is varied from 2 to 20 meters per second. Figure 4(a) shows that AODV incurs significant more control overhead (i.e., the number of control packets) than PLP-ROA. In particular, the performance of AODV deteriorates dramatically when the host mobility increases. As for the packet delivery ratio, Figure 4(b) also shows that PLP-ROA performs much better than AODV. Specifically, our technique consistently maintains a near 100% of delivery rate. In contrast, the delivery rate under AODV is about 85% at best and becomes worse and worse as the host mobility increases.

These performance results can be explained as follows. When mobile hosts move faster, a streaming

route becomes less stable, causing more frequent link breakage. Under AODV, each link breakage can directly lead to a new route discovery. While this generates a lot of control overhead, it also results in frequent stream interruption. In contrast, the proposed scheme is much less sensitive to the host mobility. Although the moving speed of mobile host increases, the average density of hosts in the network area remains unchanged. This means that the average number of hosts that can be used to construct alternative routes for critical links does not decrease. Thus, with PLP-ROA, a critical link has the same chance of being replaced by some alternative link. As a result, although Figure 4(a) shows that PLP-ROA also results in more control overhead when the host mobility increases, the stream continuity is not affected much, as indicated by the nearly constant delivery rate showed in Figure 4(b).



(a) control overhead



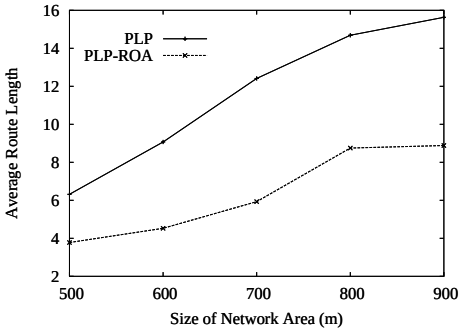
(b) data delivery ratio

In this study, we investigated the effect of host density on the performance of AODV and PLP-ROA. Again, we generated 400 mobile hosts and deployed them randomly on a network area, the size of which is varied from 500×500 to 900×900 $meter^2$. The MAX_SPEED of mobile host is set to be 10 meters per second. The results are shown in Figure 5. It shows that when host density decreases, the control overhead increases and data delivery ratio decreases. This is due to the fact that a sparse network means the distance between any two hosts is longer in average. As a result, a wireless link is more likely to be broken when the corresponding two hosts move, making a streaming route more vulnerable. In AODV, this means more route rediscovery, generating significantly more overhead and data loss. For PLP-ROA, a sparse network also means more frequently link protections. As such, it incurs more control overhead and data loss. However, the performance of this technique is much less sensitive to the host density than that of AODV, as showed in Figure 5.

In this study, we investigated how the proposed ROA technique affects the length of streaming path, which has a direct impact on the end-to-end delay of data transmission. We compared PLP-ROA against PLP, which is implemented without the ROA feature. Similar to the previous studies, we generated 400 mobile hosts and varies the network area from 500×500 $meter^2$ to 900×900 $meter^2$. The maximum speed of mobile hosts is set to be 10 meters per second. The performance results are plotted in Figure 6. Given a fixed number of mobile hosts, a larger network area means a sparse host density. Since the source and destination of a stream is randomly chosen, the stream path is longer in average under both techniques. However, by dynamically adjusting the stream path to be as straight as possible, PLP-ROA manages to maintain the length of the stream path about 50% less than PLP. This performance study convincingly shows that simply ensuring the continuity of a stream is not enough to guarantee the quality of live streaming. As the time increases, a stream path that is initially of high quality may deteriorate, making the end-to-end delay longer and longer.

4 Related Work

Routing protocols for MANET can be classified into three categories, *proactive* [9, 1], *reactive* [10, 2,



3, 11] and *hybrid* [12]. Several performance studies (e.g., [13], [14], etc.) have showed that reactive routing protocols usually have higher data delivery radio and incur less routing overhead than proactive ones. In existing reactive protocols, a host finding a broken link either finds an alternative route itself or notifies the source to initiate a route rediscovery. Since a host performs such actions only after it detects some broken link, this strategy can cause streaming interruption when applied for live streaming. Link monitoring was considered in [15] and [16], where signal strength is used to estimate a link's connection status. When a host detects that a link is about to be broken, it sends an ERR message to the source, which then initiates a new route discovery. This approach, however, can cause a lot of control overhead in the case of live streaming since each link breakage requires a rediscovery of the entire route. The PLP technique proposed in this paper can be seemed as a combination of the above two strategies.

To achieve high fault tolerance and balance traffic load balance, multi-path routing has been proposed. In [17], the multi-path routing is supported by constructing a directed acyclic graph (DAG) rooted at the destination host. In [4], two disjointed paths from source to destination are constructed simultaneously, one as a primary path and the other as a backup path. In AOMDV [5], an extension of AODV, maximal disjoint paths are discovered by asking intermediate host not to drop, but forward multiple RREQ packets as long as the new coming RREQ takes a route other than previous ones. Multi-path routing tries to redirect the network traffic to a backup path when the current path is broken. However, the backup paths may be broken as well because of host mobility. The hosts can periodically check the connectivity of the backup paths, but this would incur significant control overhead. To an extreme, it will perform just like those proactive routing protocols.

To enhance route robustness, a special multi-path protocol was recently proposed in [18]. The idea is to first find a path from sender to receiver, and then along this path, construct a mesh-like corridor. When a host receives a packet, it randomly chooses and delivers the packet to a downstream host, which is also in the corridor and closer to the destination. The downstream host repeats the same process until the packet arrives at the destination. Since a stream can flow through different paths which are not necessarily disjointed, this technique reduces the packet loss rate and may survive multiple link failures. However, when a host receiving a packet cannot find any downstream host, the link is broken and a new path needs to be found. This could happen frequently, especially when the forwarding host is on the edge of the corridor. As such, this approach shares the similar problem as other existing multi-path approaches. Another major problem of this technique is its complexity in maintaining the mesh-like corridor. Each host needs to know if it is inside the corridor, the width of which may vary as the network traffic changes; and when a host is inside the corridor, it must continuously track its neighbors, which can be expensive in reality.

5 Concluding Remarks

We have proposed a new protocol for efficient and robust live streaming in mobile ad hoc networks. The proposed technique maintains only a single routing path and has the following desirable features:

- *High continuity*: Our proactive link protection minimizes the chance of streaming interruption caused by host mobility; and in the case a streaming path is broken, it minimizes the time of discovering an alternate route.
- *Low end-to-end delay*: Our receiver-oriented adaptation makes a streaming path as straight as possible. Avoiding unnecessary detours improves the streaming efficiency and minimizes end-to-end delay.
- *Low control overhead*: Our protocol does not require periodic heartbeat from mobile hosts. In ad-

dition, both link protection and route adaptation take place locally. Thus, the control overhead is minimized

References

[1] C. E. Perkins and P. Bhagwat. Highly Dynamic Destination-sequenced distance-vector Routing (DSDV) for Mobile Computers. In *Proc. ACM SIGCOMM*, pages 234–244, 1994.

[2] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. *Mobile Computing*, pages 153–181, 1996.

[3] C. E. Perkins and E. M. Royer. Ad Hoc On-demand Distance Vector Routing. In *Proc. of IEEE WMCSA*, pages 90–100, 1999.

[4] A. Nasipuri, R. Castaneda, and S.R. Das. Performance of Multipath Routing for On-demand Protocols in Ad Hoc Networks. *ACM/Kluwer Mobile Netowrks and Applications (MONET) Journal*, pages no. 4, 339–349, April 2001.

[5] M. K. Marina and S. R. Das. On-demand Multipath Distance Vector Routing for Ad Hoc Networks. In *Proc. ICC'01*, pages 3201–3205, June 2001.

[6] S. Lee and M. Gerla. SMR: Split Multipath Routing with Maximally Disjoint Paths in Ad Hoc Networks. In *Proc. ICC'01*, pages 3201–3205, June 2001.

[7] M. R. Pearlman, Z. J. Haas, P. Sholander, and S. S. Tabrizi. On the impact of alternate path routing for load balancing in mobile ad hoc networks. In *Proc. ACM MobiHoc'00*, pages 3–10, 2000.

[8] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. A High-throughput Path Metric for Multi-hop Wireless Routing. 2003. ACM MOBICOM'03.

[9] J. Moy. OSPF version 2. *RFC 1583*, 1994. `http://www.nexor.com/public/rfc/index/rfc.html`.

[10] C. K. Toh. Associativity-based Routing for Ad-hoc Mobile Netwroks. In *Proc. of IPCCC'96*, February 1996.

[11] S. Wu, S. Ni, J. Sheu, and Y. Tseng. Route Maintenance in Mobile Ad Hoc Network. *Telecommunication System, Kluwer Academic Publishers*, pages 61–84, 2001.

[12] Z. J. Haas, M. R. Pearlman, and R. Samar. The Zone Routing Protocol (ZRP) for Ad Hoc Networks. July 2002. `http://www.ietf.org/proceedings/02nov/I-D/draft-ietf-manet-zone-zrp-04.txt`.

[13] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva. A Performance Comparison of Multi-hop Wireless Ad Hoc Network Routing Protocols. In *ACM MOBICOM'98*, pages 85–197, October 1998.

[14] S. R. Das, C. E. Perkins, and E. M. Royer. Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks. In *Proc. of the IEEE Conference on Computer Communications*, pages 3–12, March 1999.

[15] T. Goff and N. B. Abu-Ghazaleh etc. Preemptive Routing in Ad Hoc Networks. In *Proc. 7th Conf. Mobile Computing and Networking*, pages 43–52, July 2001.

[16] L. Qin and T. Kunz. Pro-active Route Maintenance in DSR. In *Mobile Computing and Communication Review*, pages 79–89, July 2002.

[17] V. D. Park and M. S. Corson. A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks. In *Proc. of IEEE INFOCOM'97*, pages 1405–1413, 1997.

[18] F. Dai and J. Wu. Proactive Route Maintenance in Ad Hoc Networks. In *Proc. IEEE ICC'05*, May 2005.